# A Web Predictive Model Based on Dependency Graph

Bin Liu
Shengda Economics Trade & Management College, Zhengzhou University, Zhengzhou, China 451191
Email: liu_bin1221@yahoo.com.cn

Haizhen Shi, Lei Shi
School of Information Engineering, Zhengzhou University, Zhengzhou, China 450001
Email: shz418@163.com, shilei@zzu.edu.cn

Lin Wei
School of Software, Zhengzhou University, Zhengzhou, China 450002
Email: weilin@zzu.edu.cn

Zhanhong Wang[1, 2]
[1]School of Information Engineering, Zhengzhou University, Zhengzhou, China 450001
[2]School of Computer and Information Technology, Xinyang Normal University, Xinyang, China 464000
Email: yywmb@yahoo.com.cn

*Abstract*—**The essence of Web prefetching is the construct of prediction algorithm and prefetching control. And the heart of Web prefetching is the prediction model, which is mainly classified into two main categories according to the data structure taken into account to make the prediction, namely PPM model and DG model. Although PPM model has higher predictive accuracy, it occupies large storage space. Although DG prediction algorithm can overcome PPM's shortcoming, it can't distinguish the initial weight of different order and do not take into account how web pages are structured, so its prediction accuracy is lower. Therefore, an improved prediction model is proposed to remedy the above defects. Experimental results show that the improved model can save the storage space and enhance higher prediction precision for the pages which have a high amount of embedded objects.**

*Index Terms*—**Web prefetching, Web prediction, DG model, PPM model**

## I. Introduction

In recent years, the web has become the primary means for information dissemination. It is being used for commercial, entertainment, or educational purposes and, thus, its popularity resulted in heavy traffic in the Internet. Since the Internet capacity is not keeping pace, the net effect of this growth was a significant increase in the global traffic, damaging the quality of service(availability, reliability, security) and specially the user perceived latency, that is, the time between when a client issues a request for a document and the time the response arrives. Potential sources of latency are the web servers' heavy load, network congestion, low bandwidth, band width underutilization and propagation delay.

Corresponding author, Prof. Lei Shi, Zhengzhou University

The problem of modeling and predicting a user's accesses on a web-site has attracted a lot of research interest. It has been used to improve the web performance through caching and prefetching, recommend related pages, improve search engines and personalize browsing in a web site.

The caching of web documents at various points in the network (client, proxy and server) has been developed to reduce latency. Caching capitalizes on the temporal locality. Effective client and proxy caches reduce the client perceived latency, the server load and the number of traveling packets, thus increase the available bandwidth. Nevertheless there exist cases where the benefits reaped due to caching can be limited, e.g., when web resources tend to change very frequently, resources cannot be cached (dynamically generated web documents), when they contain cookies (this issue matters only caching proxies), or when request streams do not exhibit high temporal locality.

Web prefetching techniques, known as active caching techniques, by analyzing the user's access history, preprocess the user's requests before the user demands them explicitly and get the objects into the cache. For users, the prefetching techniques mask the server processing time and network transfer delaying time. The key of prefetching techniques is to establish an effective prediction model and then make an accurate prediction based on the prediction model.

Pandey [1] and Nanopoulos [2], based on previous work, find that the majority of prediction model can be classified into two categories: PPM (Prediction by Partial Match) prediction model which is based on tree structure; DG (Dependency Graph) prediction model which is based on graph structure. In recent years, there have been many researches working on the Web prefetching techniques. Prediction by Partial Match (PPM) is a commonly used technique in web prefetching, where prefetching decisions are made based on historical URLs in a dynamically maintained Markov prediction tree [3]. Guaranteed the accuracy, most of the researches focus on the reducing of the space occupation of the PPM prediction model because of its large space complexity. For the typical PPM prediction model, there are the standard PPM and the LRS (Longest Repeating Sequences) PPM.

The DG prediction algorithm constructs a dependency graph that depicts the pattern of accesses to the objects. The graph has a node for every object that has ever been accessed. As defined by Griffioen [5] in file systems prediction, and implemented by Padmanabhan [4] in web prefetching, there is an arc from one object to another object. The weight of the arc is the access probability from one object to another object. When the client request arrives, the server dynamically updates the model, and predicts the next request.

PPM prediction algorithm has better prediction accuracy, but it takes enormous space complexity. Although DG prediction algorithm can overcome PPM's shortcomings, it can't distinguish the initial weight of different order and do not take into account how web pages are structured [6], so its prediction accuracy is lower.

Based on the above discussion, we propose an improved algorithm based on exponential descendent double dependency graph algorithm. By analyzing the user's access history and the relationship between web objects, according to the features of the web surfing and user access depth, we make use of exponential descendent algorithm to make up for defects of DG model. Accordingly, it not only makes the prediction more precise, but also reduces the space complexity.

## II. RELATED WORK

Users go surfing usually among web documents through a hyperlink. Such browsing behavior implies common access patterns and laws. These laws not only determine the popularity of visited pages, but also determine the probability distribution of user browsing depth (the number of pages which user accessed in a web site). By studying the web surfing features and user access behavior, we will optimize the DG to control the scale of model in breadth and in depth. In breadth we discuss about the Zipf's law, and in depth we take into account Gaussian distribution.

First, studies show web objects with high frequency are applicable to Zipf's first law. When $a$ ( $a$ is a constant close to 1) increases, web pages with higher frequency have higher possibility to be accessed [8]. Zipf's first law shows that most requests only access a small part of pages, while the requests are extremely sparse for most of the pages.

For low-frequency web objects, Zipf's first law has larger deviation. Accordingly, Zipf proposes Zipf's second law for low-frequency words. According to the Zipf's second law, we can estimate that the number for the web pages visited once to the total page number is 50-60%, which is 17% for the web pages visited twice and 8.3% for the web pages visited three times. This phenomenon indicates that the low-frequency Web objects takes up a large number of requests.

TABLE I.
SOME PAPERS SUPPORTING ZIPF'S LAW

| Objects which obey Zipf's distribution | References |
|---|---|
| Web object | Steven Glassman. A caching relay for the world wide web. 94 |
| | Carlos Cunha, et.al. Characteristics of WWW client-based traces. 95 |
| | Virgilio Almeida, et.al. Characterizing reference locality in the www. 96 |
| | L. Breslau,P.Cao, et.al. Web caching and Zipf-like distributions. 99 |
| | B. A. Huberman and L. A. Adamic. Growth Dynamics of the world wide web. 99 |
| | D.N.Serpanos, G.KaraKostas. Effective caching of web objects using Zipf's law. 2000 |
| | G.Karakostas, D.N.Serpanos. Exploitation of different types of locality for web caches. 2002 |
| | Lei Shi. Web caching and prefetching research based on the popularity. 2006 |
| | Toshihiko Yamakami. A zipf-like distribution of popularity and hits in the mobile web pages with short life time. 2006 |
| Stream media | Chesire Maureen, et.al. Measurement and analysis of a streaming-media workload. 2001 |
| | Cherkasova Ludmila, Minaxi. Characterizing locality, evolution, and life span of accesses in enterprise media server workloads.2002 |
| | Tang W, Fu Y, et.al.MediSyn: a synthetic streaming media service workload generator. 2003 |
| | Guo.L. DISC. Dynamic interleaved segment caching for interative streaming. 2005 |
| | Di Wu, Yong Liu and Keith W. Ross. Modeling and analysis of multiChannel P2P live video systems. 2010 |

In some papers, they show that web requests follow Zipf's law. Several authors have applied Zipf's law to the distribution of web requests and stream media. In table I show several main papers to support this claim. By using Zipf's law, it can help us to effectively choose which object need to cache. Caching high-frequency objects will greatly enhance the hit ratio.

However, Guo [9] represents the reference rank distributions cannot be fitted with a straight line in a log-log scale, meaning they are not Zipf-like. Instead, they follow a distribution which is called a stretched exponential distribution (SE). For stream media, caching of media workload (SE) is far less efficient than that of Web (Zipf) workload. For example, assuming all objects are cacheable and have the same file size, caching 1% Web content can only achieve 18% hit ratio, even though they have the same hit ratio with an unlimited cache. Furthermore, with a much higher temporal locality, long-term caching can have a high hit ratio greater than 85% with caching 10% content.

In table II, some researches show that objects don't follow Zipf's law. In recent researches, from the model point of view, actual streaming data is generally consistent with Zipf-like model.

From the supported Zipf's law thesis or not, we can clearly recognize: low-frequency objects which account for a higher proportion can be removed without affecting the overall situation; high-frequency objects which account for a lower proportion will affect the final results. Therefore, we can get a conclusion: high-frequency objects can affect the final result, however, the low-frequency objects which is few contributions or no contributions for the hit ratio can be used as a basis for pruning. Caching high-frequency objects will greatly enhance the hit ratio.

Next, the number of links a user follows before the page value first reaches the stopping L threshold is a random variable. The probability distribution of first passage times to a threshold is given asymptotically by the two parameter inverse Gaussian distribution [7].

$$P(L) = \frac{\sqrt{\lambda}}{\sqrt{2\pi L^3}} \exp\left[-\frac{\lambda(L-\mu)^2}{2\mu^2 L}\right] \tag{1}$$

With mean E (L) $= u$ and variance Var [L] = $u^3/\lambda$. L (user browsing depth) is the number of links that a user accessed in a Web site.

This distribution has two characteristics worth stressing in the context of user surfing patterns. First, it has a very long tail, which extends much further than that of a normal distribution with comparable mean and variance. This implies a finite probability for events that would be unlikely if described by a normal distribution. Consequently, large deviations from the average number of user clicks computed at a site will be observed. Second, because of the asymmetry of the distribution function, the typical behavior of users will not be the same as their average behavior. Thus, since the mode is lower than the mean, care has to exercise with available data on the average number of clicks, as it overestimates the typical depth being surfed.

TABLE II.
SOME PAPERS NOT SUPPORTING ZIPF'S LAW

| Objects which don't obey Zipf's distribution | References |
|---|---|
| Web objects | Virgilio Augusto F, et.al. Analyzing the behavior of a proxy server in the light of regional and cultural issues. 98 |
| | Norifumi Nishikawa, et.al. Memory-based architecture for distributed WWW caching proxy. 98 |
| Stream media | Acharya S, Smith B C, Parnes P. Characterizing user access to videos on the world wide web.99 |
| | Chu J, Labonte K, BN. Availability and locality measurements of peer-to-peer file systems. 2002 |
| | Gummadi KP, Dunn RJ, et.al. Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. 2003 |
| | Bellissimo, Levine, et.al. Exploring the use of BitTorrent as the basis for a large trace repository. 2004 |
| | Klemm A, Lindemann C, et.al. Characterizing the query behavior in peer-to-peer file sharing systems. 2004 |

User browsing characteristics provide a theoretical basis for cutting down the vertical scale of the model, that is, user's browsing depth is not messy but laws and relatively concentrated. Experimental results show that the expectation of user browsing depth is located in between 3 and 6. In order to maintain higher prediction accuracy and control the scale of model, improved DG model dynamically calculates the E (L) as the value of the greatest lookahead window.

## III. DG ALGORITHM

The prediction of users' accesses has been usually made by adapting prediction algorithms from other fields of computing to deal with web accesses. For instance, the DG algorithm, which was firstly used in web prefetching by Padmanabhan, lies in a predictor of accesses to a local file system. PPM algorithm, mainly used for lossless data compression, has been also used to predict web accesses by several authors. With these algorithms, web prefetching achieved an acceptable performance when they were proposed. However, the Web has changed noticeable during the last decade. There are two main differences between old and current Web generations: the increasing dynamism and customization of the content, and the increase in the complexity of the web site design. Despite this fact, prefetching algorithms taking into account the new Web sites characteristics have not been proposed.

With the advent of web2.0, there is a noticeable increase in the amount of embedded objects per page. In this context, most of the predictions made by the existing algorithms are useless to reduce the user-perceived latency because these algorithms do not take into account how web pages are structured, i.e., an HTML object with several embedded objects. These predictions are useless since the client already knows that the following objects to be demanded are the embedded objects. However, these objects are not still demanded because the amount of simultaneous connections to a web server from the same client is limited, as suggested in the standard HTTP/1.1 and implemented in the most commonly used web browsers. Therefore, there is neither time to prefetch nor perceived latency to reduce.

Also, DG algorithm doesn't distinguish the initial weight of different order. In general, one order should have a larger weight than other orders. The weight of different order should be different.

In this part, we first introduce the DG algorithm. In the next chapter, we present the improved DG prediction algorithm that considers the characteristics of the current web sites and distinguishes the initial weight of different order to improve the performance achieved by web prefetching.

First, we consider a web log as a relation table T that is defined by a set of attributes $A = \{A1, A2, ..., Am\}$. Usual attributes include Host, Ident, Authuser, Time, Request, Status, Bytes, Referrer, Agent and Cookie. Assume that transactions generated by different users are identified by a subset of attributes $S \subset A$. Let $u$ be a set of user ids and $F : S \rightarrow U$ a function that maps each unique combination of values of $S$ to a user id of $U$. Let $At \notin S$ be the Time attribute.

**Definition 1.** A session s is an ordered set of transactions in $T'$ which satisfy $A_U(t_{l+1}) = A_U(t_l)$ and A $A_t(t_{l+1}) - A_t(t_l) < \tau$ where $t_{l+1}$ , $t_l \in s$ and $\tau$ is a given time threshold (30 minutes).

Conceptually, a session defined in Definition 1 is a set of ordered request objects viewed in one visit by the same visitor. In a session, these objects have the order before and after. What's more, the types of these objects may be different.

Given the above considerations, we define a DG model as follows:

**Definition 2.** A DG model is a $\langle S,O,T,W \rangle$ where S, O, T, W are the sets of indices for four dimensions (Session, Objects, Types, Window) in which

1. S indexes all identified sessions s1, s2,…, sn,
2. O indexes all identified objects o1, o2,…, on,
3. T is a type of the object where we can divide the object into two types which are html object and image object,
4. W is the lookahead window size.

**Definition 3.** Let F be a mapping from $\langle S,O,T,W \rangle$ to M that performs that we enter a session, a request object sequence, the type of the object and the size of w, so we can get a matrix.

M=F (S, O, T, W ) where s∈S, o∈O, t∈T, and w∈W.

Assume the sequence of the session s1 is H1, Ob1, H2, Ob2, H3 and Ob3; the size of w is 2.Therefore we have

$$M(s1) = \begin{pmatrix} 0 & 1 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0.5 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Where 1 in the first row and second column stands for the weight from H1 to Ob1; 0.5 in the first row and third column stands for the weight from H1 to H2; 0 stands for two objects not unreachable. If we enter another session, so we need to update the matrix. It must be explained that in the matrix the size of w is fixed and we don't consider the type of the object in order to describe questions from mathematical model.

From the intuitive point of view, we can construct a dependency graph by the rules as follows:

1) The graph has a node for each URL that has ever been accessed. It uses a structure which stores the number of visited nodes. For example, A (2) means that node A is accessed for twice.

2) In the graph, there is an arc from node A to B if and only if at some point in time a client accessed to B within w accesses after A, where w is the lookahead window size.

3) The weight of the arc is the ratio of the number of accesses to B within a window after A to the number of accesses to A itself. Within w, for i<w the initial weight of arc set 1; the confidence of each arc is calculated by dividing the counter of the arc by the amount of appearances of the node.

Figure 1 shows the state of the DG algorithm after a training example with the following accesses: H1, Ob1, H2, Ob2, H3, Ob3 by one user; and H1, Ob1, H4, Ob2, H5, Ob3 by other user.

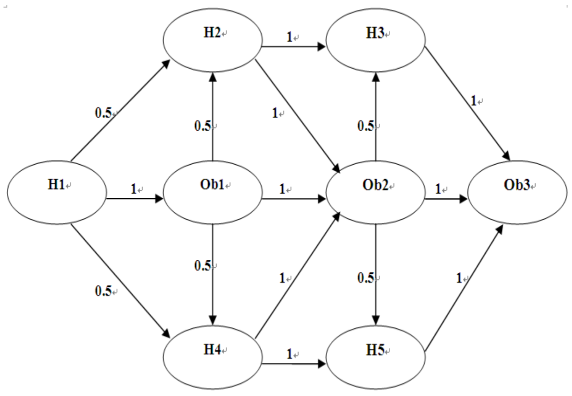


Figure 1. DG.

The advantage of DG model is easy to achieve, but it can't distinguish the initial weight of different order and do not take into account how web pages are structured, so predictive results are not obvious.

## IV. Improved DG Algorithm

On the one hand, according to characteristics of web object popularity and Zipf's second law, improved model removes the noise objects whose weight is less than the threshold θ. On the other hand, according to inverse Gaussian distribution of user browsing depth, improved model sets the value of lookahead window w by dynamically computing the expectation of inverse Gaussian distribution. And improved DG model distinguishes the initial weight of different order by using the exponential descent algorithm which set the weight of arc to be 1/2i. Therefore, improved DG model can further improve the prediction accuracy.

Improved DG prediction algorithm constructs a dependency graph by the rules as follows:

1) The graph has a node for each URL that has ever been accessed. It uses a structure which stores the number of visited node. For example, A (2) means that node A is accessed for twice.

2) In the graph, there is an arc from node A to B if and only if at some point in time a client accessed to B within w accesses after A, where w is the lookahead window size. But unlike DG, it distinguishes two classes of dependences: dependences to an object of the same page and dependences to an object of another page. The arc is a primary arc if A and B which are objects of different pages. On the contrary, the arc is secondary arc.

3) The weight of the arc is the ratio of the number of accesses to B within a window after A to the number of accesses to A itself. Within w, for i<w the initial weight of arc set 1/2i, the confidence of each arc is calculated by dividing the counter of the arc by the amount of appearances of the node, both for primary and for secondary arcs.

Figure 2 shows the state of the DG algorithm after a training example with the following accesses: H1, Ob1, H2, Ob2, H3, Ob3 by one user; and H1, Ob1, H4, Ob2, H5, Ob3 by other user. Arrows with continuous lines represent primary arcs while dashed lines represent secondary arcs. Primary and secondary arcs represent the relation between object accesses of different pages and between object accesses of the same page, respectively.

The predictions are obtained firstly by applying a cutoff threshold to the weight of the primary arcs that leaves from the node of the last user access. In order to predict the embedded objects of the following page, a secondary threshold is applied to the secondary arcs that leave from the nodes of the objects predicted in the first step.

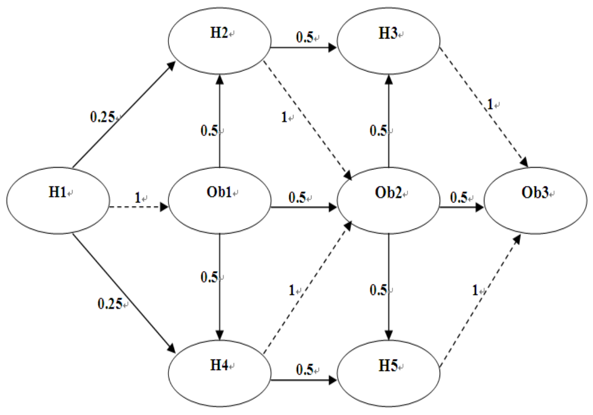


Figure 2. Improved DG model.

For example, the confidence of the arc of node Ob1 to node Ob2 when using DG prediction algorithms:

$$P\ (Ob1, Ob2) = (1+1)\ /2= 1 \qquad (2)$$

The confidence of the arc of object Ob1 to object Ob2 when using improved DG prediction algorithms:

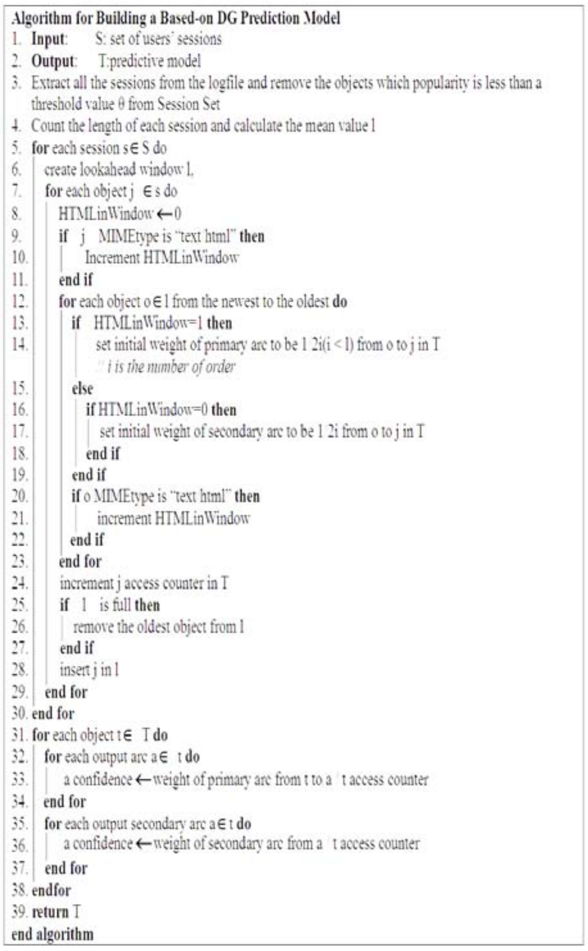
$$P\ (Ob1, Ob2) = (1/2+1/2)\ /2= 0.5 \qquad (3)$$

The pseudo code for building the improved DG prediction model is shown in Figure 3.

## V. THE CACHING ALGORITHM

However, there are cases where a non-effective prefetching algorithm has some drawbacks, can impact cache performance. What's more, caching scheme performs better in high density mobile networks than in sporadic circumstance [10]. For instance, if the accuracy of the prefetching algorithm is low, then several useful documents in the cache may be evicted by prefetched documents that are not going to be referenced. Therefore, in this paper, we will put the prefetched documents into a dedicated part of the cache, to avoid the drawback of incorrect replacement of requested documents.

The caching algorithm PECache [11] divides the cache into a Weighing Room (uses LRU as the replacement policy) and a Waiting Room (uses the FIFO policy). This division of the cache space aims at isolating the effect of document mispredictions or the effect of aggressive prefetching. It achieves this by dedicating part of the cache space to exploit the temporal locality of the request stream (on-demand request) and the rest of the cache space is dedicated to exploit the spatial locality (prefetch requests). The relative size of the partitions should reflect the amount and type of the locality of the request stream.



```
Algorithm for Building a Based-on DG Prediction Model
1. Input:    S: set of users' sessions
2. Output:   T:predictive model
3. Extract all the sessions from the logfile and remove the objects which popularity is less than a
   threshold value θ from Session Set
4. Count the length of each session and calculate the mean value l
5. for each session s∈S do
6.   create lookahead window l.
7.   for each object j ∈s do
8.     HTMLinWindow ←0
9.     if j MIMEtype is "text html" then
10.      Increment HTMLinWindow
11.    end if
12.    for each object o∈l from the newest to the oldest do
13.      if HTMLinWindow=1 then
14.        set initial weight of primary arc to be 1 2i(i < l) from o to j in T
             // i is the number of order
15.      else
16.        if HTMLinWindow=0 then
17.          set initial weight of secondary arc to be 1 2i from o to j in T
18.        end if
19.      end if
20.      if o MIMEtype is "text html" then
21.        increment HTMLinWindow
22.      end if
23.    end for
24.    increment j access counter in T
25.    if l is full then
26.      remove the oldest object from l
27.    end if
28.    insert j in l
29.  end for
30. end for
31. for each object t∈ T do
32.   for each output arc a∈ t do
33.     a confidence ←weight of primary arc from t to a  t access counter
34.   end for
35.   for each output secondary arc a∈t do
36.     a confidence ←weight of secondary arc from a  t access counter
37.   end for
38. endfor
39. return T
end algorithm
```

Figure 3. Pseudo code for building the improved DG prediction model.

The caching procedure PECache (Prefetch Enhanced Cache) has as input the requested document (d) and the current request stream of the user (R) as follows:

```
Procedure PECache(Array R, Document d)
begin
1. R=RUd
2. if not (d in Weighing Room or d in Waiting Room)
2. put d at head of the LRU list of the Weighing Room
3. prefetchSeq=Prefetch(R, M, maxSize)
4. foreach p in prefetchSeq
5. append p at the end of Waiting Room queue
6. endfor
7. else if d in Waiting Room
```

```
8. remove d from Waiting Room
9. put d at head of the LRU list of the Weighing Room
10. else if d in Weighing Room
11. put d at head of the LRU list of the Weighing
    Room
12. endif
end
```

## VI. Experimental Evaluation

This section presents the experimental results. In order to test the performance of improved DG algorithm, we use a real web log to predict the performance. Experiment data derives from Inventive Information Technology Co. E-commerce Platform log files [12] and Henan University log files [13]. For DG model and improved DG model, the experiment uses 1/2 of log files as a training set and the remaining serve as a test set.

Three performance key metrics are used to evaluate web prefetching benefits:

**Precision (Pc)** measures the ratio of good prediction to the number of prediction. Precision, defined in this way, just evaluates the algorithm without considering physical system restrictions; e.g., cache, network or time restrictions; therefore, it can be seen as a theoretical index.

$$Pc = \frac{Good\ Predictions}{Predictions} \quad (4)$$

**Recall (Rc)** measures the percentage of user requested objects that were previously prefetched. Recall quantifies the weight of the predicted (see 5) or prefetched (see 6) objects over the amount of objects requested by the user. Notice that this index only deals with the number of good predictions made but not with the total amount of predictions.

$$Rc = \frac{Good\ Predictions}{User\ Requests} \quad (5)$$

$$Rc = \frac{PrefetchHits}{User\ Requests} \quad (6)$$

**Traffic overhead** measures the ratio of cache misses re-request cost bandwidth and prefetch bandwidth-consuming to total bandwidth.

$$T = \frac{BandWidth^{+} + BandWidth^{-}}{BandWidth_{all}} \quad (7)$$

From the figure 4, in prediction we can see that when w is 1, DG model and improved DG model have the same prediction accuracy. With the increasing of w, prediction accuracy of improved DG model significantly has been enhanced. When w is 3, prediction accuracy of improved DG is steady. And when w continues to increase, the accuracy is not obviously enhanced. But for the DG model, when w is 2, the prediction accuracy reaches its maximum. Later with the increasing of w, the accuracy is worse. The reasons for this phenomenon are that DG model can't distinguish the initial weight of different orders. When w is too large, it actually reduces the weight of first order. So when w is increasing, it will appear that the prediction accuracy declines. What's more, figure 5 shows that the precision and recall of improved DG has been enhanced higher than one in the figure 4. Because the former has higher amount of embedded objects per page, for example, in e-commerce platform there are a large amount of images. As for recall, improved DG model also has a relatively better performance.

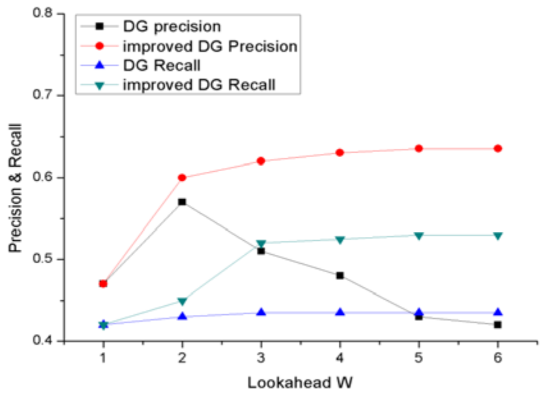


Figure 4. Henan University log

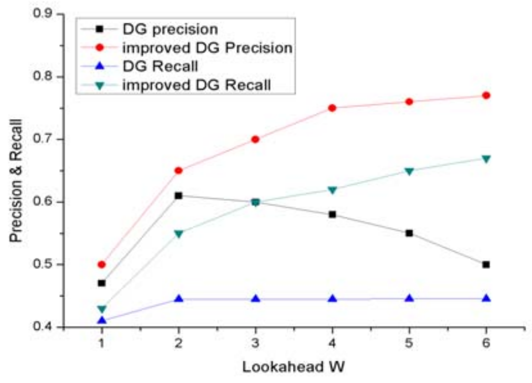


Figure 5. Inventive e-commerce platform log

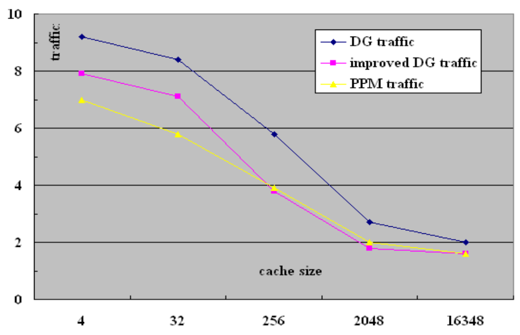


Figure 6. Henan University log

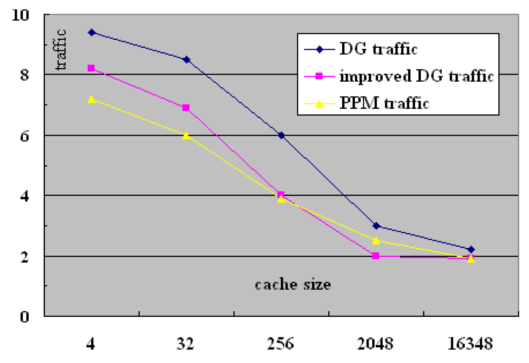


Figure 7. Inventive e-commerce platform log

Figure 6 and figure 7 shows that DG, improved DG and PPM are compared in the traffic overhead as a cost factor, when the lookahead w is 3. Experiment results show that: when the cache is fewer, due to the need to be frequently replaced for the objects in the cache, thus the traffic overhead is higher. With the cache size increasing, traffic overhead is significantly decreased.

Figure 8 shows that improved DG and PPM are compared in the precision and recall, when the lookahead w is 3. From the picture, we can find that improved DG obtains the similar performance in precision and recall as the PPM.

In storage space, figure 2 shows improved DG model only stores 8 nodes information where PPM model need to store 38 nodes information in figure 9. Therefore, improved DG algorithm can greatly reduce the space complexity of the model.

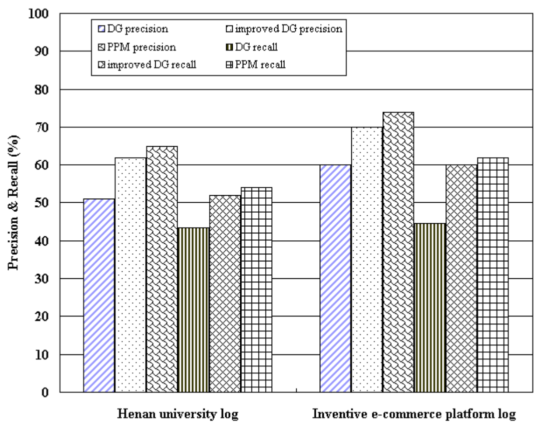


Figure 8. Precision & recall

## VII. CONCLUSIONS AND FUTURE WORK

For the two categories of web prediction model, in recent years, the problem of modeling and predicting a user's accesses on a web-site by PPM has attracted a lot of research interest; however, the research about the DG is relatively less.

In this paper, according to web surfing features and user browsing depth, we improved and optimized the DG prediction model. What we have done is to distinguish the initial weight of different order by using the exponential descent algorithm which set the initial weight of arc to be 1/2i. In contrast to existing algorithm, experimental results show that the model achieves certain effectiveness in improving prediction accuracy and reducing space complexity.

In the future, we will test and analyze the performance of our approach by establishing a unified evaluation model. And we will combine our algorithm with practical application in order to adapt actual work better.

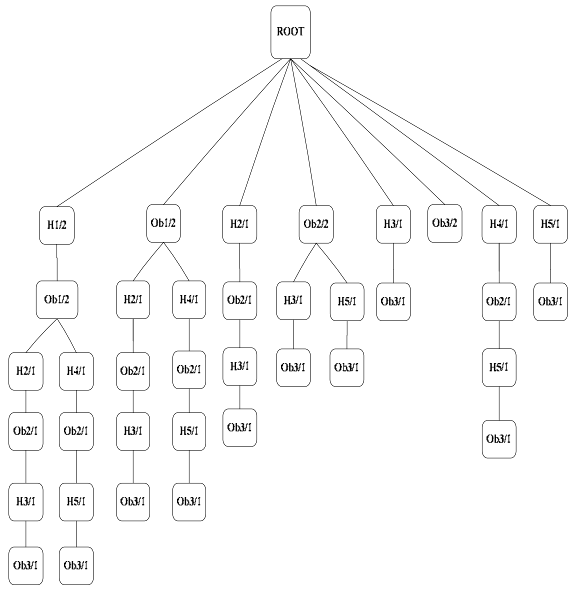


Figure 9. PPM model

## REFERENCES

[1] J. Pandey, A. Goel, "A framework for predictive Web prefetching at the proxy level using data mining," International Journal of Computer Science and Network Security, 2008, vol. 8(6), pp. 303−308.

[2] A. Nanopoulos, D. Katsaros, "A data mining algorithm for generalized web prefetching," IEEE transactions on knowledge and data engineering, 2003, vol. 5(5), pp. 1155−1169.

[3] Lei Shi, Yingjie Han, Xiaoguang Ding, et.al, "An SPN based Integrated Model for Web Prefetching and Caching," Journal of Computer Science and Technology, 2006, vol. 21(4), pp. 482−489.

[4] V. N. Padmanabhan, J.C. Mogul, "Using predictive prefetching to improve World Wide Web latency," Computer Communication Review, 1996, vol. 26(3), pp. 22−36.

[5] J.Griffioen, R. Appleton, "Reducing file system latency using a predictive approach," In Proc. of 1994 Summer USENIX conference, 1994. pp. 197−207.

[6] Josep Domenech, Jose A. Gil, "DDG: An efficient prefetching algorithm for current Web generation," In proceedings of the 1st IEEE workshop on hot topics in Web systems and technologies (HotWeb), Boston, USA, 2006.

[7] B. Huberman, P. Pirolli, J. Pitkow, and R. Lukose, "Strong regularities in World Wide Web surfing," Science 280 (5360), pp. 95−97, 1998.

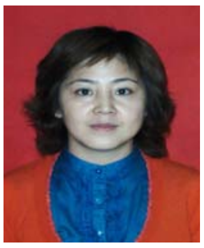
[8] E P Marcatos, C E Chronaki, "A Top-10 approach to prefetching the Web," Proceedings of the Eighth Annual Conference of the Internet Society, 1998.

[9] L. Guo, E. Tan, S. Chen, Z. Xiao, and X. Zhang, “Does Internet media traffic really follow Zipf-like distribution?” In Proc. of ACM SIGMETRICS, 2007.
[10] Xiaohui Chen, Weidong Wang, Guo Wei, “Performance of Web Caching in High Density Mobile Networks,” IEEE international Conference on Computer and Information Science, 2009.
[11] C.Umapathi, J.Raja, “A prefetching algorithm for improving web cache performance,” Journal of Applied Sciences, 2006, pp. 3122–3127.
[12] http://www.inventinfotech.com.
[13] http://www.henu.edu.cn.

**Bin Liu** received the M. Sc. degree (2003) in Multi-media and virtual reality from LEUVEN GROUPT College. Now he is a lecturer in Shengda Economics Trade & Management College, Zhengzhou University. His research interests are in the areas of distributed and Internet systems, 2D-3D image convert and transmit, web mining and high graphic performance computing.

**Haizhen Shi** received the B.Sc. degree (2008) in computer science from Huanghe Science and Technology University. Currently he is a M.Sc. candidate in computer application technology in the School of Information Engineering, Zhengzhou University. His research interests include different areas in web prefetching, web mining and E-commerce.

**Lei Shi** received the M.Sc. degree (1992) in computer science from Nanjing University and Ph.D. degree (2006) in computer science from Beijing Institute of Technology. He has published over 70 papers in journals and international conferences and 5 books in his research field. More than 20 papers are indexed by SCI, EI and ISTP. Currently he is a full professor in the School of Information Engineering, Zhengzhou University. He served as workshop chair, program committee member of many international conferences. His research interests are in the areas of distributed and Internet systems, web prefetching, web mining and high performance computing.



**Lin Wei** received the M.Sc. degree (2006) in computer science from Zhengzhou University. Currently she is an associate professor in the School of Software, Zhengzhou University. Her research interests include web prefetching, web mining and E-commerce.

**Zhanhong Wang** was born in Kaifeng, Henan, China, in September 1979. He received his B.Sc. degree from Xinyang Normal University, China, in 2003. He is now a M.Sc. candidate of Zhengzhou University. His research interests include distributed Internet systems, web mining and web prefetching.